1 Task description

This project address the possibility to automatically select leaning experiences that facilitate the development of general solutions. The outcome of an adaptive process is strongly influenced by the conditions experienced during robots' evaluation. In standard approach this problems is normally approached by selecting the initial environmental conditions randomly. This method is simple and can support the development of solutions that are robust with respect to environmental variations. However, when infrequent conditions required qualitatively different treatments than more frequent conditions, the adaptive process might converge on solutions optimized for frequent conditions only.

THE OBJECTIVE of this project is that to implement and evaluate a selection process that use the information extracted from previous evaluations to select environmental conditions requiring different treatments.

THIS CAN be realized by:

- generating a set of N different learning conditions at the beginning the the training process,
- using an NxN matrix to store the difference in performance achieved in each possible couple of conditions,
- by using these data to select environmental conditions requiring different treatments during multiple evaluation episodes.

The matrix can be updated on the basis of the performance achieved by the top individuals of each generation over multiple evaluation episodes. Environmental conditions enquiring different treatments can be chosen by selecting the first condition randomly, and by using the matrix to identify conditions that produced different result from the already chosen conditions The data contained in the matrix can can also be used to select conditions that have an intermediate level of difficulty, on the average. In other words conditions that are neither too easy neither too difficult at the current stage of the evolutionary process.

The approach can be validated for example on the double pole balancing problem.

2 Model and statement of the problem

The pole balancing system has one or more poles hinged to a wheeled cart on a finite length. The movement of the cart and the pole is constrained within a two-dimensional plane. The objective of the problem is to balance the poles infinitely by applying a force to the cart at a regular time intervals, such that the cart stays within the track boundaries.

A double pole balancing problem consists of the cart that is free to travel along a linear rail, with two parallel pendulums in dangling from the cart. Pendulums are fixed on the center of

the cart and don't affect each other. The only actuator is a force input into the cart parallel to the rail - the pendulums are completely underactuated. This model has three degrees of freedom and only one active linear actuator. The movement space of the cart is limited by borders. The schematic view of the double pole balancing problem is presented in figure 1.

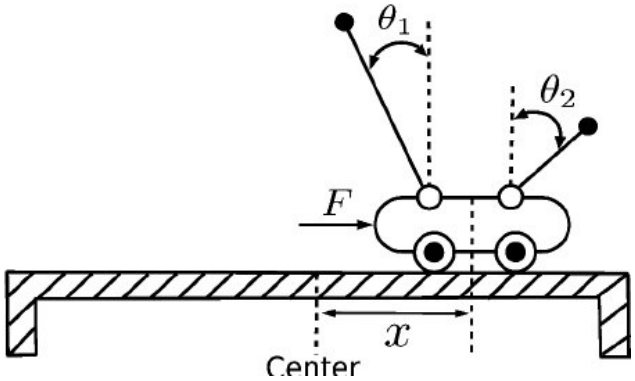


Figure 1: The double pole balancing problem

The poles must be balanced simultaneously by applying the continuous force F to the cart. The parameters x, θ_1, and θ_2 are the offset of the cart from the center of the track, and the angles from the vertical of the long and short pole, respectively.

The balance system can be described by the following dynamic equation:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{\theta_1} \\ \ddot{\theta_2} \end{bmatrix} + n = F = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u \tag{1}$$

where:

$$M = \begin{bmatrix} M + m_1 & L_1 m_1 cos(\theta_1) & L_2 m_2 cos(\theta_2) \\ L_1 m_1 cos(\theta_1) & L_1^2 m_1 & 0 \\ L_2 m_2 cos(\theta_2) & 0 & L_2^2 m_2 \end{bmatrix} \tag{2}$$

$$n = \begin{bmatrix} -L_1 m_1 sin(\theta_1) \dot{\theta_1}^2 - L_2 m_2 sin(\theta_2) \dot{\theta_2}^2 \\ -m_1 g L_1 sin(\theta_1) \\ -m_2 g L_2 sin(\theta_2) \end{bmatrix} \tag{3}$$

The x, θ_1 and θ_2 is system variables, M is mass of cart, m_1 and m_2 is mass of pendulums, L_1 and L_2 is a length of the poles, respectively, and g is a gravity coefficient. The system can be modelled by the state space vector:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{\theta_1} \\ \dot{\theta_2} \\ M^{-1}(\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T u - n) \end{bmatrix} \tag{4}$$

where $X = \begin{bmatrix} x & \theta_1 & \theta_2 & \dot{x} & \dot{\theta}_1 & \dot{\theta}_2 \end{bmatrix}^T$
The system parameters for this model are:

- $M = 1\ kg$,
- $m_1 = 0.1\ kg$,
- $m_2 = 0.05\ kg$,
- $L_1 = 0.5\ m$,
- $L_2 = 0.25\ m$,
- $g = 9.8\ \frac{N}{kg}$.

The challenge of the problem is to keep both pendulums upright for the duration of the simulation. The track boundary conditions for this task are:

- the track is limited by edges equaled to 2.4 m of each side of the center of the rails,
- each angle can not be more the 60°.

Each of these conditions stops the simulation, and the task is considered a failure.

3 Environment Simulation

Dynamical simulators are necessary for two main purposes. They are

- simulation of the dynamics of articulated robots,
- simulation of the effect of collisions.

However, in the case of wheeled robots moving on a flat surface, kinematic simulations can be sufficient to use. Kinematic simulations allow reducing its cost comparing to the dynamical simulations. The evorobotpy library presents a set of the environments involving wheeled robots that rely on kinematic simulations implemented in C++.

One of these environments is called ERDpole, described as an environment with a robot. The robot is a wheeled cart with two poles on its center. Figure 2 presents the simulation of the environment. The task to be solved consists of balancing the poles around an equilibrium position. The environment is perfect for simulation of the double pole balance problem that it is a task of the project.

However, for solving the main task of the project, especially, to implement and evaluate the selection process of environmental conditions, several changes should be processed. As the

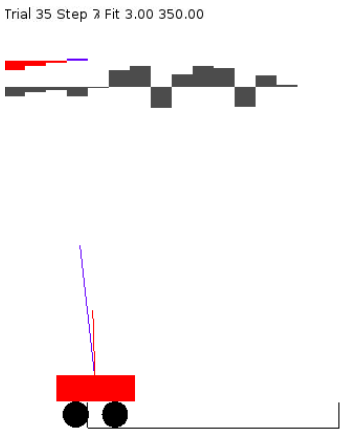


Figure 2: ErDpole environment

task description suggests, at the beginning of the initialization of the program, that evolves the double balancing problem, it generates the matrix of environmental conditions. The robot trains to balance the poles via 6 environmental conditions. They are:

- an initial position of the cart,
- the velocity of the cart,
- an inclination of the first pole,
- a velocity of the first pole,
- the inclination of the second pole,
- the velocity of the second pole.

The construction of the matrix assumes the values for each environmental condition. Each condition considers the next values:

- First condition - $[-1.944, -0.972, 0, 0.972, 1.944]$,
- Second condition - $[-1.215, -0.6075, 0, 0.6075, 1.215]$,
- Third condition - $[-0.10472, -0.05236, 0, 0.05236, 0.10472]$,
- Forth condition - $[-0.135088, -0.067544, 0, 0.067544, 0.135088]$,
- Fifth condition - $[-0.10472, -0.05236, 0, 0.05236, 0.10472]$,

- Sixth condition - $[-0.135088, -0.067544, 0, 0.067544, 0.135088]$.

The size of the generated matrix is $[15625 \times 6]$, where each row represents the state of 6 environmental conditions with all combinations. On the initialization step, the program loads the matrix to select one of the condition sets. The aim is to train the robot to balance the pole on the chosen states of the environment.

Attaching the matrix to the program impacts on additional changes in the system, particularly in choosing the row with a set of conditions of the matrix. Formerly, during the evolutionary stage, the algorithm Salimans generate the random seed of states to train to balance the poles. However, to solve the task at every evolutionary stage of training as initial conditions, the states should be taken from the matrix the same. That is why the seed initialization changes the Salimans algorithm.

Due to 15625 variations, a great amount of time spent on a procession of the combination of every state without additional changes in the source code. That is why the project proposes the usage of parallel computations in the evolutionary stage at the seeds and saving data to the file without conflicts among the parallel process.

4 Assessment of the initial state difficultness

For an assessment of difficulty, we use the two types of metrics. The first metric considers the assumption. If the initial state of the system is closer to the state of the equilibrium and the cart has more place for maneuver, then the probability is higher to complete the task throughout the entire simulation time. The second metric considers the potential and kinetic energy of variables of the model.

Since the equilibrium state located at the zero position than we can describe this set of metrics by the following way:

$$k_1 = x^2; \quad k_2 = \theta_1^2; \quad k_3 = \theta_2^2; \quad k_4 = (\theta_1 - \theta_2)^2 \tag{5}$$

$$Difficultness = k_1 + k_2 + k_3 + k_4 \tag{6}$$

the k_4 metric considers the assumption. If two pendulums are on opposite sides of the equilibrium position, then the task becomes more complicated as the difference increases. The metric comes from the equations of system dynamics. In any case, if pendulums are on opposite sides of the cart, then one of them will fall due to the inertia created by the cart.

The second type of metric describes through the following equations. The equations present

the interaction of the kinetic and potential energy:

$$E_1 = \frac{1}{2}M\dot{x}^2; \quad E_2 = 0 \tag{7}$$

$$E_3 = \frac{1}{2}m_1(L_1^2\dot{\theta_1}^2 + 2cos(\theta_1)L_1\dot{\theta_1}\dot{x} + \dot{x}^2); \quad E_4 = m_1gL_1cos(\theta_1) \tag{8}$$

$$E_5 = \frac{1}{2}m_2(L_2^2\dot{\theta_2}^2 + 2cos(\theta_2)L_2\dot{\theta_2}\dot{x} + \dot{x}^2); \quad E_6 = m_2gL_2cos(\theta_2) \tag{9}$$

$$E = E_1 + E_2 + E_3 + E_4 + E_5 + E_6 \tag{10}$$

We can estimate how the initial state for the model is difficult, especially how much the model quickly learns from the proposed data and what maximum fit we can get. The impact of each of these metrics considers learning outcomes.

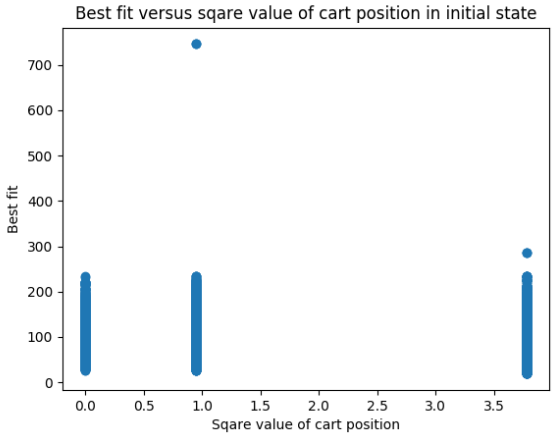


Figure 3: Graph of relation between training results and k_1 metric

As you can see in figure 3, the initial position of the cart does not greatly affect the learning outcome. But at the same time, we want to note that the maximum fit achieves at a distance that is not large to the center position of the cart.

Figure 4 shows the first and second graphs. The graphs show the maximum fit reached in the position close to the equilibrium. The assumption reports that finding two poles on opposite sides of the equilibrium position greatly complicates the task. Regarding the third graph, the assumption becomes true. The system learns more difficult when the difference between the first and second pendulum increases.

Let's consider the energy metrics of the system. In figure 5, you can see the results of the calculation energy of system parts relative to the best fit.

As you can see, the initial potential energy of the cart does not affect the learning outcomes in any way, since it is always zero. As for the kinetic energies of the trolley and pendulums, the smaller they are, the higher the probability that the model will fulfill the task.

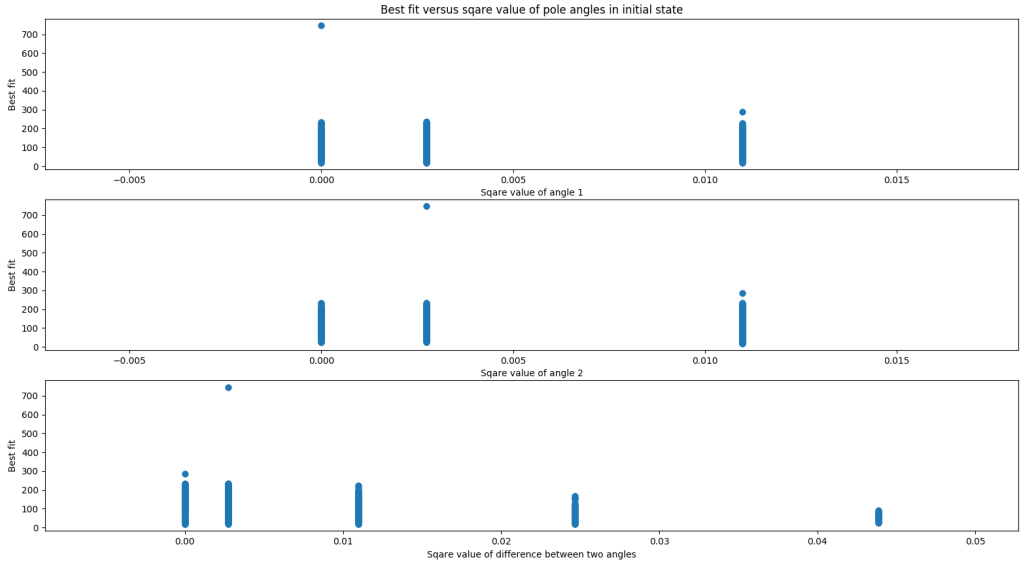


Figure 4: Graph of relation between training results and k_2, k_3 and k_4 metrics

The reverse situation is with the potential energy of the poles. The potential energy for poles depends on the angle of the inclination of the poles. The smaller it is, the greater the potential energy is. According to the process of the task performing, the algorithms should strive to maximize the potential energy of the poles transferring from the kinetic energy. As you can see in the graphs, the high initial potential energy of the poles leads to a higher chance of completed tasks for the entire duration of the simulation.

Based on the analysis of two metrics, we identified 9 clusters of varying complexity. You can see their visualization in figure 7.

The complexity of each cluster distributes in decreasing order of energy and increasing order of $Difficultness$ metric. This means that the simplest initial states belong to cluster 6, and the most complex ones belong to cluster 1. Since the potential energy of poles is higher than the initial kinetic energies are.

The figure shows the 3D graph of the dependencies of learning outcomes on the metrics of complexity and energy. From the figure, you can see the sequence of the increasing complexity of the initial state from each cluster.

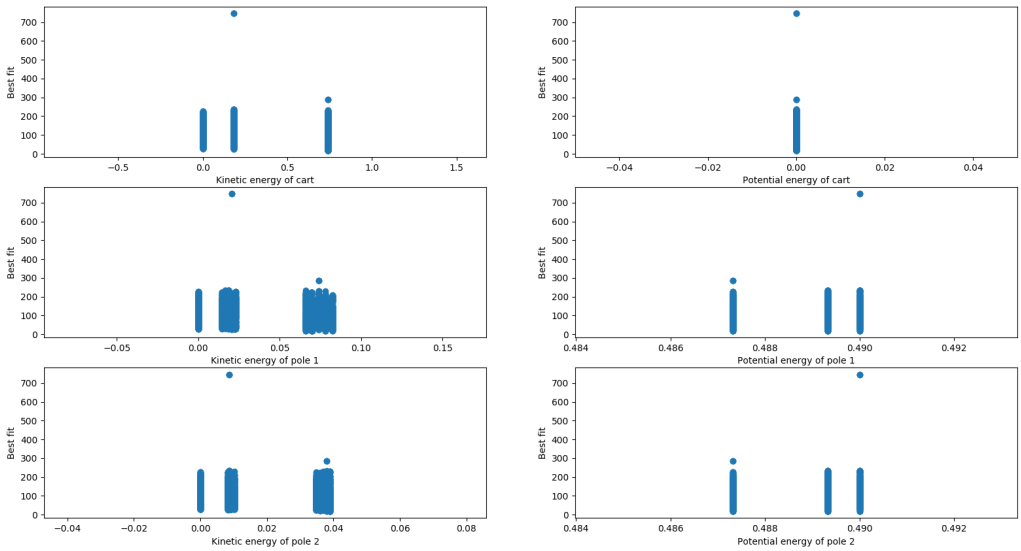


Figure 5: Graphs of relation between train results and energy of system parts

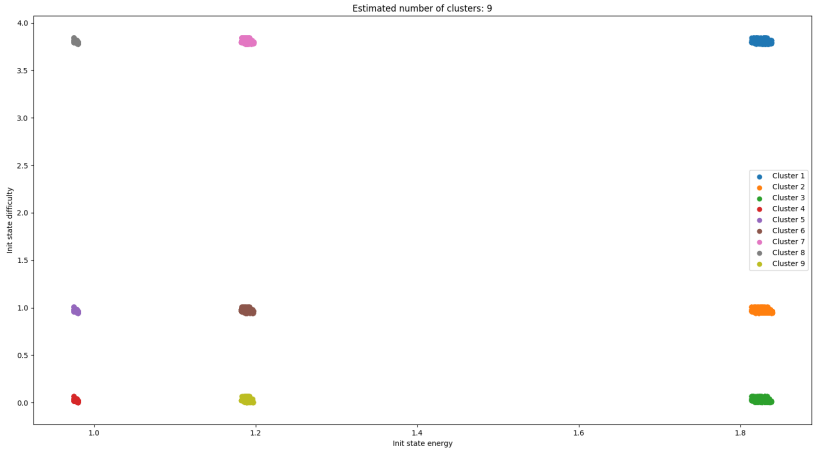


Figure 6: Clasterization of initial state difficulty

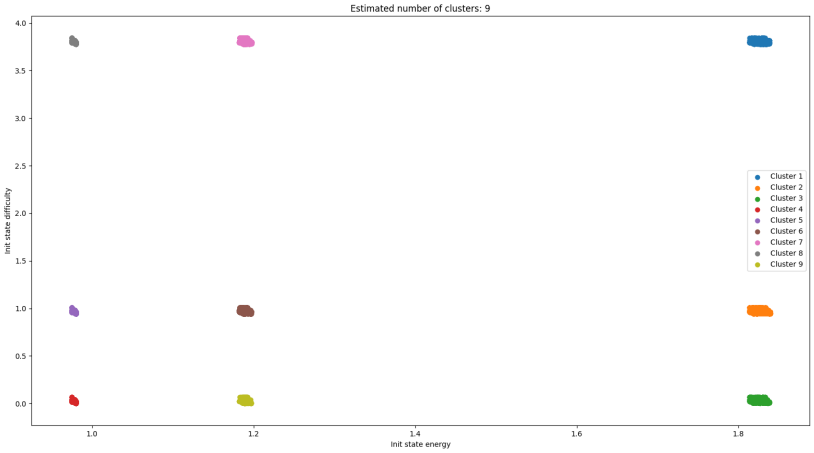


Figure 7: Clasterization of initial state difficulty

5 Consequences of diversified conditions choices

To analyze the consequences of choosing the diversified conditions, firstly, it should be considered by what metric or estimation of evolutionary stages, the analysis of the described system is processed. However, before explaining why and which metrics the analysis uses, the evolutionary process should be described.

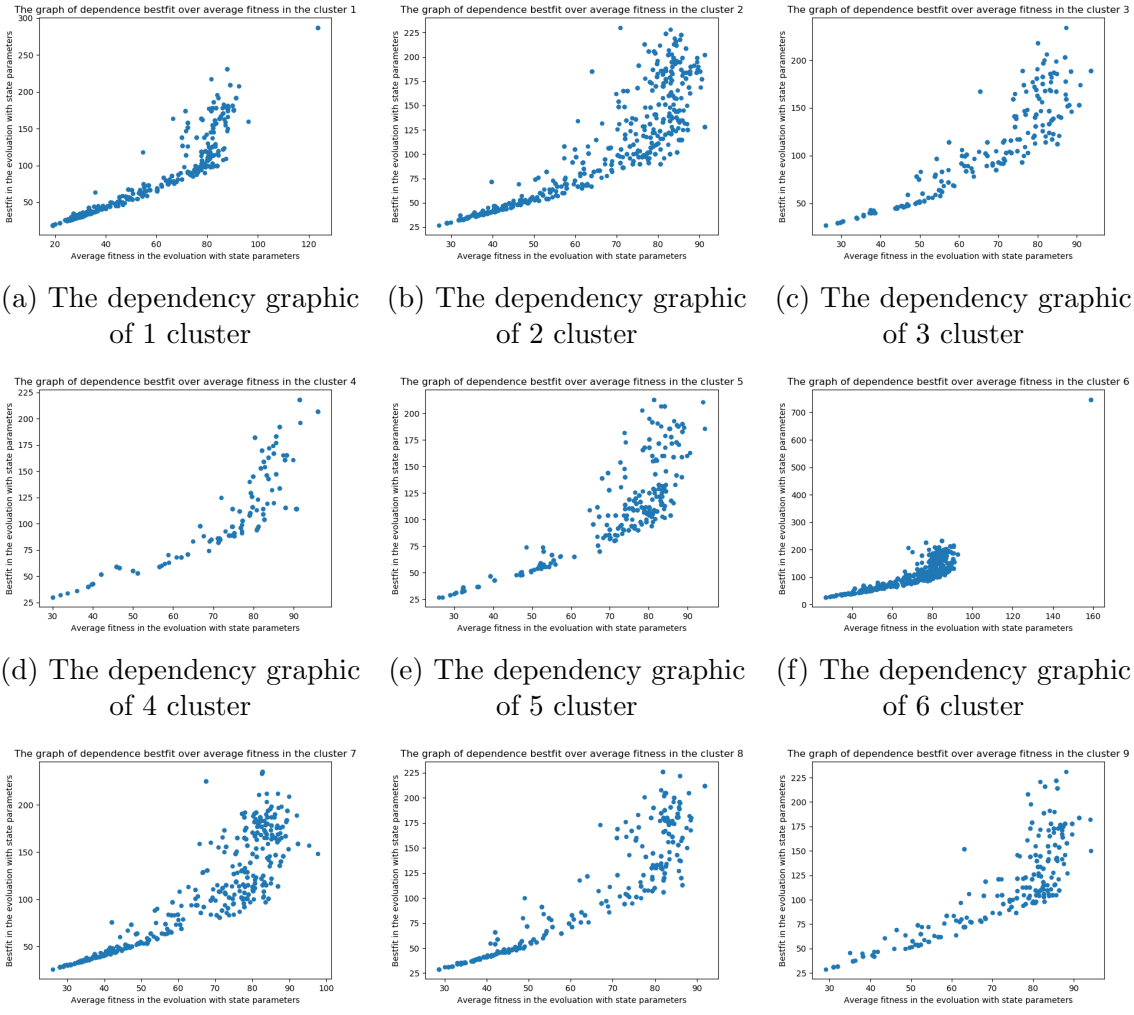
Evolutionary processes can be defined as follows:

- First of all, to the system, the random seed is given as an input to choose the conditions in the matrix. More precisely, the seed represents the index of the row of the matrix.
- Secondly, the evolutionary stages begin, where the robot learns to balance the poles at every stage with the same conditions chosen at the initial step.
- Thirdly, at each evolutionary stage, the fitness is calculated as rewording for balancing the poles.
- Finally, at each seed (evolutionary conditions of the robot), the program generates a list of fitness at every evolutionary stage. Due to the list, the program computes the best and average fitness.

The parameters best and average fitness are taken as parameters to analyze the consequences of diversified conditions choices. The average fitness shows the mean value of all fitnesses at evolutionary stages as the central tendency of the data. The best fitness shows the maximum value at evolutionary stages. That is why it is concerned to show the difference of maximum

fitness from the central tendency of the data depending on the chosen environmental conditions.

As described in section 4, the trained data can be divided into 9 clusters by the difficulty and the energy. We consider analyzing the fitness in the 9 clusters. Figure 8 presents the dependency graphics of best fitness over average fitness at each cluster.



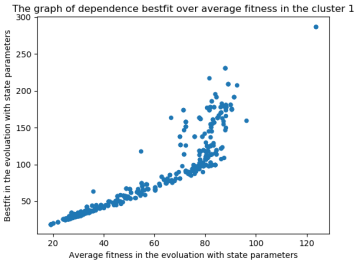
(a) The dependency graphic of 1 cluster (b) The dependency graphic of 2 cluster (c) The dependency graphic of 3 cluster

(d) The dependency graphic of 4 cluster (e) The dependency graphic of 5 cluster (f) The dependency graphic of 6 cluster

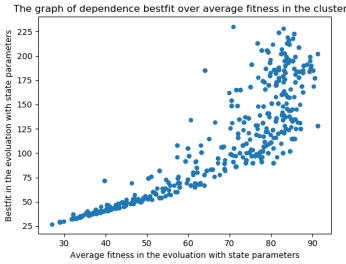
(g) The dependency graphic of 7 cluster (h) The dependency graphic of 8 cluster (i) The dependency graphic of 9 cluster

Figure 8: The dependency charts of the best fitness over average

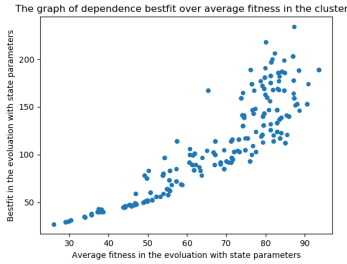
For further analysis, it is convenient to rearrange charts by difficulty. Figure 9 shows the following graphics of clusters represented by the simplistic environmental conditions. In figure 10, the charts present the dependency in clusters of medium environmental conditions by difficulty. In figure 11, the charts present the dependency in difficult clusters.



(a) The dependency graphic of 1 cluster

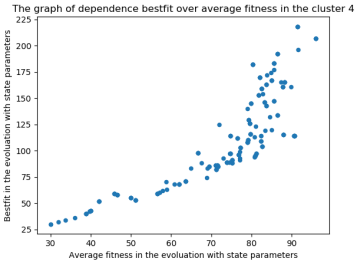


(b) The dependency graphic of 2 cluster

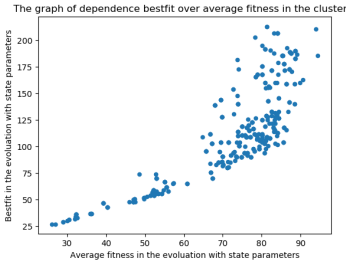


(c) The dependency graphic of 3 cluster

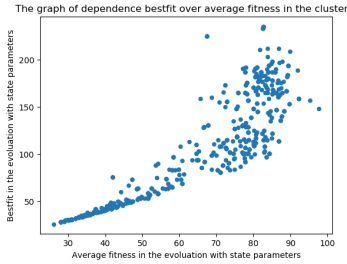
Figure 9: The dependency charts of the best fitness over average in simple environmental conditions



(a) The dependency graphic of 4 cluster

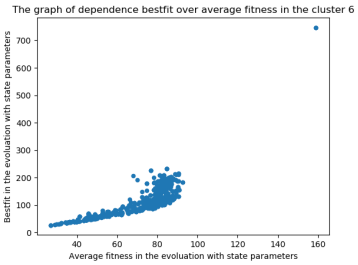


(b) The dependency graphic of 5 cluster

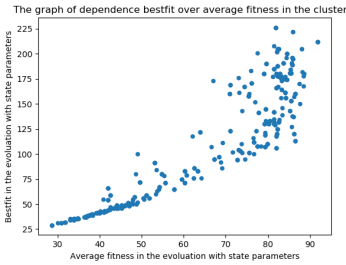


(c) The dependency graphic of 7 cluster

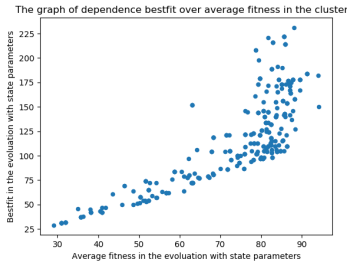
Figure 10: The dependency charts of the best fitness over average in medium environmental conditions



(a) The dependency graphic of 6 cluster



(b) The dependency graphic of 8 cluster



(c) The dependency graphic of 9 cluster

Figure 11: The dependency charts of the best fitness over average in difficult environmental conditions

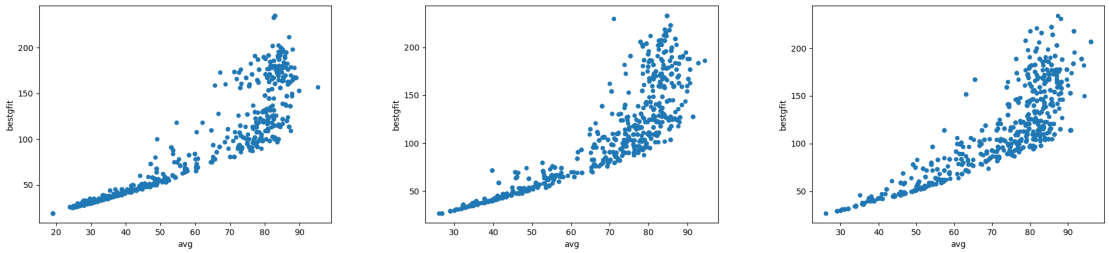
From the charts, the behavior of fitness at each cluster is the same. In some environmental states, the best fitness is near to the central tendency at some too far from it. However, at

the 1 cluster, 2, and 3, that is considered as the simplest, the graph can be approximate by linear regression with a small set of outliers that form a cloud in on the upper right part of the graph. At the complicated clusters, such as 9, 6, and 7, the linear regression also can be processed. But there exists a great number of outliers that form a cloud with high density on the upper right part of the graph. At the medium difficult environmental conditions, that form 4, 5, and 8 clusters, the linear regression approximates and cloud exists. But the density of points in the clouds and linear regression are approximately the same.

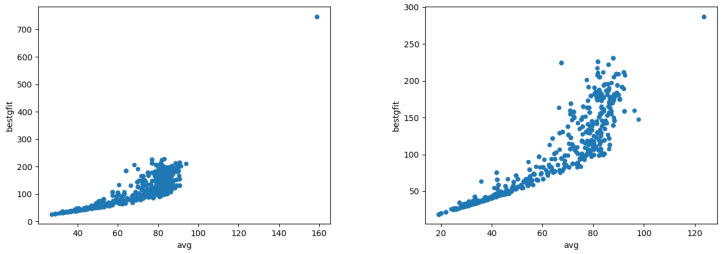
From the analyzed results it can be concluded that with increasing difficultness in environment conditions increase the frequency of the high difference between best fitness and average fitness at each evolutionary processes.

To analyze the consequences of choosing the diversified environmental conditions from a certain state it should be considered the dependencies of the fitness at each state.

Figure 12 presents the charts that show the dependency of fitness at the fixed environmental condition called the initial position of the cart at every considered value for the condition.



(a) Initial position = -1.944 (b) Initial position = -0.972 (c) Initial position = 0

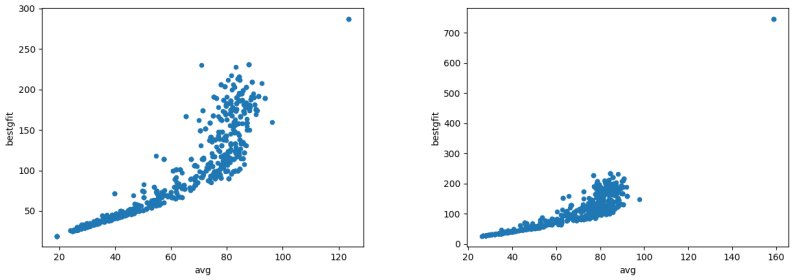


(d) Initial position = 0.972 (e) Initial position = 1.944

Figure 12: The dependency charts of the best fitness over average at the fixed environmental condition - the initial position of the cart

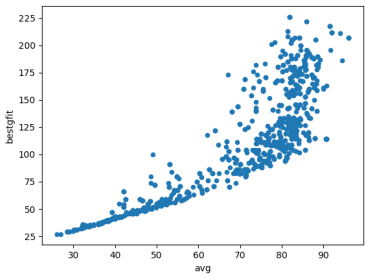
The behavior of the graphic is the same as in the previous graphics divided into clusters. It means that fixing the first state does not bring any new behavior to the graphics. However, the points location and the distribution density are different at the fixed initial position of the cart. The maximum value of the best fitness on each graph is around 200 - 250, but as you can see on the 12d chart the maximum fitness reaches the value of more than 700.

Figure 13 presents the charts that show the dependency of fitness at the fixed environmental condition called the velocity of the cart at every considered value for the condition.

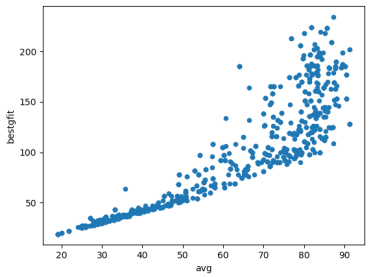


(a) Cart velocity = -1.215

(b) Cart velocity = -0.6075



(c) Cart velocity = 0

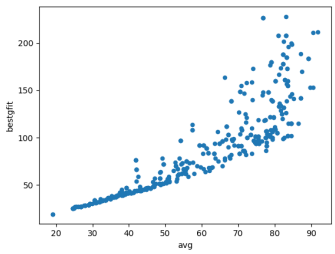


(d) Cart velocity = 1.215

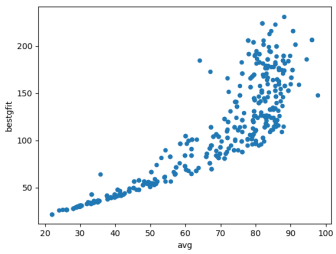
Figure 13: The dependency charts of the best fitness over average at the fixed environmental condition - the velocity of the cart

The behavior of the graphic is the same as in the previous graphics. It means that fixing the second state also does not bring any new behavior to the graphics. The pattern on each chart is almost the same. Although, on figures 13a and 13b, the maximum value of fitness reaches around 300 and more than 700 points, respectively. As you can see, on the figure 13 the chart with fixed state cart velocity equaled to 0.6075 does not provide, because due to computation time, the program estimates not all states.

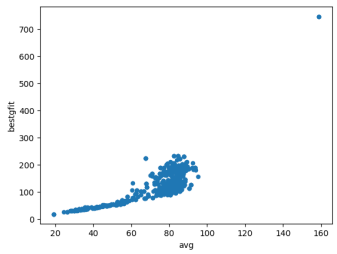
Figure 14 presents the charts that show the dependency of fitness at the fixed environmental condition called the inclination of the first pole at every considered value for the condition.



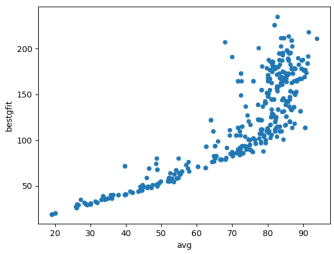
(a) Inclination of first pole = -0.10472



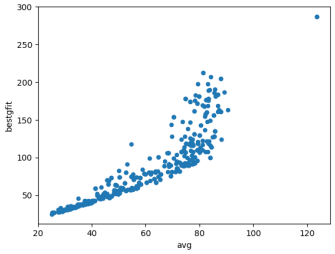
(b) Inclination of first pole = -0.05236



(c) Inclination of first pole = 0



(d) Inclination of first pole = 0.05236

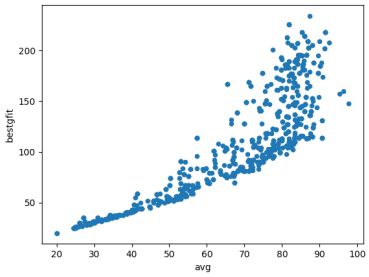


(e) Inclination of first pole = 0.10472

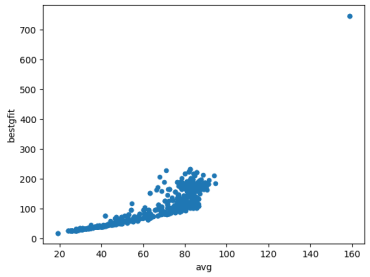
Figure 14: The dependency charts of the best fitness over average at the fixed environmental condition - the inclination of the first pole

The behavior of the graphic is the same as in the previous graphics. It means that fixing the inclination of the first pole also does not provide any new behavior to the graphics. The pattern on each chart almost the same. Still, the density of points is more in forming approximately a linear regression than a cloud in the upper right part of the chart. However, on figures 14e and 14c the maximum value of fitness reaches around 300 and more than 700 points respectively.

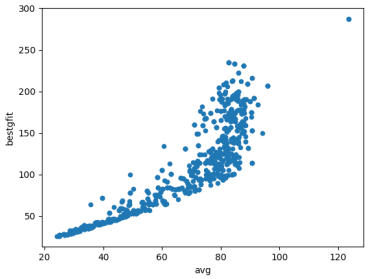
Figure 15 presents the charts that show the dependency of fitness at the fixed environmental condition called the velocity of the first pole at every considered value for the condition.



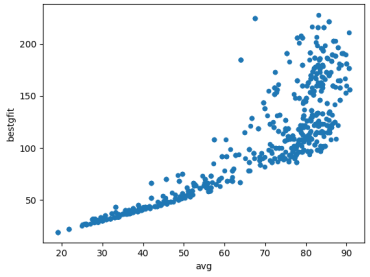
(a) First pole velocity = -0.135088



(b) First pole velocity = -0.067544



(c) First pole velocity = 0

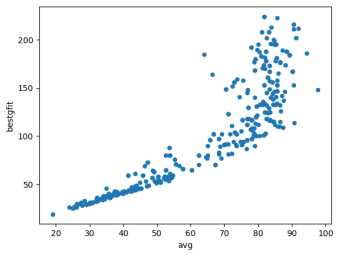


(d) First pole velocity = 0.135088

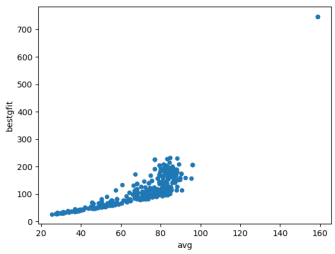
Figure 15: The dependency charts of the best fitness over average at the fixed environmental condition - the velocity of the first pole

The behavior of the graphic is the same as in the previous graphics. It means that fixing the velocity of the first pole environmental state does not bring any new behavior to the graphics. However, the pattern of charts is slightly different from the previous one. The density of points spreads almost evenly. In figures 15c and 15b, the maximum value of fitness reaches around 300 and more than 700 points respectively. The previous graphics show the same behavior with fixed condition states. As you can see, on the figure 15 the chart with fixed state first pole velocity equaled to 0.067544 does not provide, because due to computation time, the program estimates not all states.

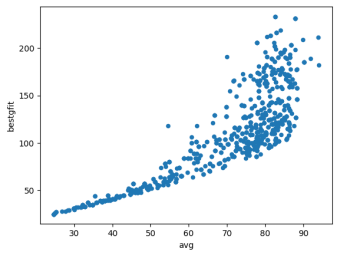
Figure 16 presents the charts that show the dependency of fitness at the fixed environmental condition called the inclination of the second pole at every considered value for the condition.



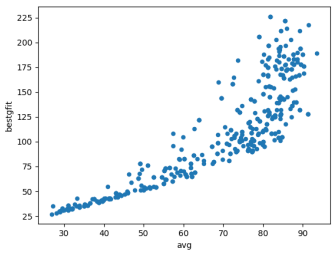
(a) Second pole inclination = -0.10472



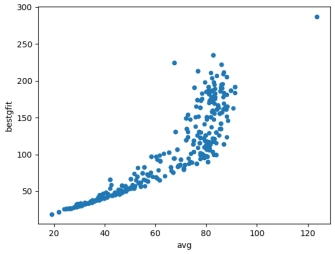
(b) Second pole inclination = -0.05236



(c) Second pole inclination = 0



(d) Second pole inclination = 0.05236

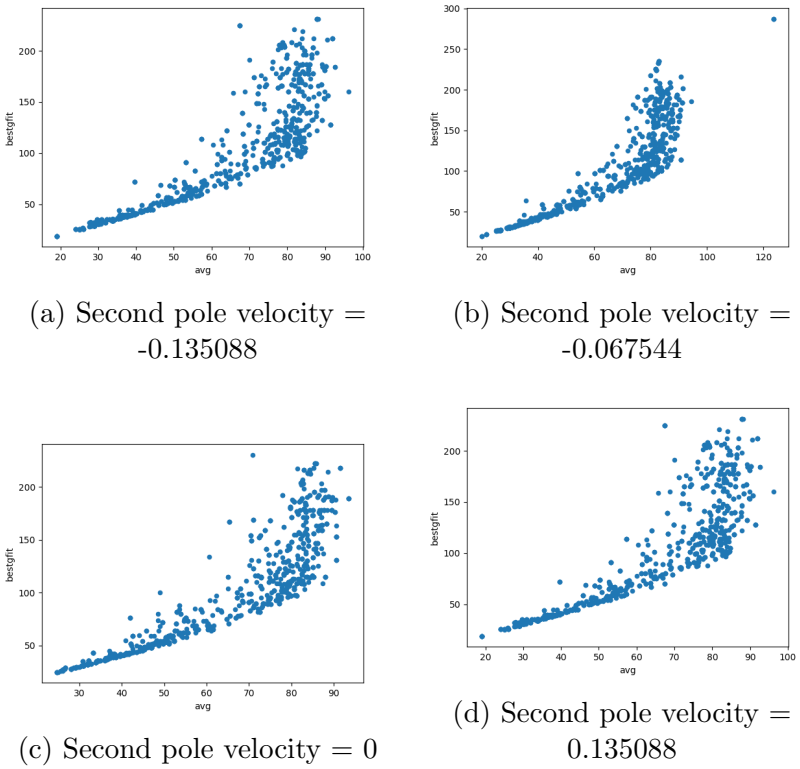


(e) Second pole inclination = 0.10472

Figure 16: The dependency charts of the best fitness over average at the fixed environmental condition - the inclination of the second pole

The behavior of the graphic is the same as in the previous graphics. It means that fixing the fifth environmental state does not bring any new behavior to the graphics. However, the pattern of charts is slightly different from previous and different to charts itself. The density of points spreads not evenly. In some figures, the density of points is high on the left part of the graphic, like 16a. In some figures, the density is high on the right part of the chart, like on 16c. Or the density spreads evenly as on 16d. In figures 16e and 16b, the maximum value of fitness reaches around 300 and more than 700 points respectively. The previous graphics show the behavior with fixed condition states.

Figure 17 presents the charts that show the dependency of fitness at the fixed environmental condition called the velocity of the second pole at every considered value for the condition.



(a) Second pole velocity = -0.135088

(b) Second pole velocity = -0.067544

(c) Second pole velocity = 0

(d) Second pole velocity = 0.135088

Figure 17: The dependency charts of the best fitness over average at the fixed environmental condition - the velocity of the second pole

The behavior of the graphic is the same as in the previous graphics. It means that fixing the sixth state also does not bring any new behavior to the graphics. The pattern on each chart is almost the same. Although, in figures 17b, the maximum value of fitness reaches around 300 points. As you can see, on the figure 17, the chart with fixed state second pole velocity equaled to -0.067544 does not provide, because due to computation time, the program estimates not all states.

6 Conclusion

In the solution of the project task, we implement the automatic selection of learning experience from the extraction of information from previous evaluations to select environmental conditions. For that purpose, we use the environment conditions matrix that holds all different states combinations.

In the learning step, the program chooses the initial states from the matrix and perform evolutionary learning to balance the poles. Moreover, the difficulty and consequences of the environment conditions choices were evaluated and analyzed in this report.

There were found the clusters of difficulty and evaluated the consequences on them.